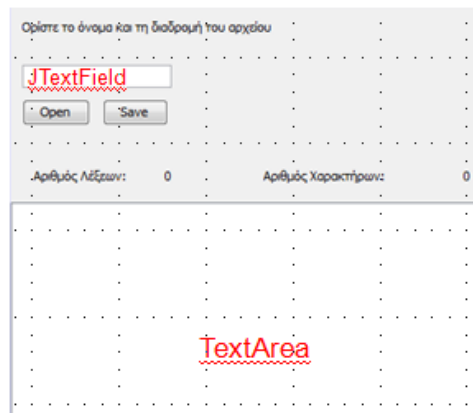
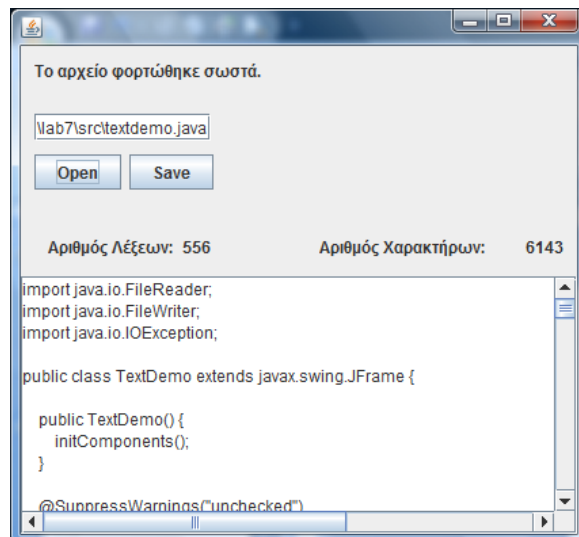


Εργαστήριο 7 - Άσκηση - Ανάλυση

Εκφώνηση: Δημιουργήστε την εφαρμογή «simple editor» σε Java Swing με χρήση NetBeans. Στην εφαρμογή αυτή ο χρήστης θα μπορεί να φορτώνει, αποθηκεύει απλό αρχείο κειμένου και να εμφανίζει τον αριθμό των χαρακτήρων και των λέξεων που περιέχονται μέσα στο αρχείο. Ακολουθείστε τις παρακάτω οδηγίες:

1. Η εφαρμογή θα σχεδιασθεί σε ένα εξωτερικό υποδοχέα JFrame, ο οποίος θα έχει την παρακάτω εμφάνιση:



2. Για τη λειτουργία του open θα πρέπει να κάνετε τα παρακάτω:
 - a. FileReader fr; (για να δημιουργήσετε αντικείμενο FileReader ώστε να διαβάσετε από αρχείο)
 - b. TextArea.getText(); (για να διαβάσετε και να μεταφέρετε όλο το κείμενο του textarea σε κατάλληλη μεταβλητή).

- c. Εμφανίστε κατάλληλο μήνυμα στην ετικέτα πάνω από τα κουμπιά όταν
 - i. δεν έχει εισαχθεί όνομα και διαδρομή αρχείου, δηλαδή όταν το JTextField πάνω από τα κουμπιά είναι άδειο,
 - ii. έχει εισαχθεί λάθος όνομα αρχείου,
 - iii. ολοκληρωθεί επιτυχημένα το άνοιγμα αρχείου.
 - d.

```
fr = new FileReader(όνομα αρχείου και διαδρομή);
jTextArea1.read(fr, null);
fr.close();
```

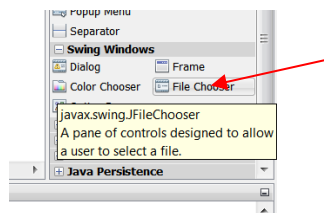
Χρησιμοποιήστε τον παραπάνω κώδικα για να εμφανίσετε τα περιεχόμενα ενός αρχείου στο TextArea.
3. Για τη λειτουργία του save θα πρέπει να κάνετε τα παρακάτω:
 - a. FileWriter fw; (για να δημιουργήσετε αντικείμενο FileWriter ώστε να γράψετε σε αρχείο)
 - b. Εμφανίστε κατάλληλο μήνυμα στην ετικέτα πάνω από τα κουμπιά όταν
 - i. δεν έχει εισαχθεί όνομα και διαδρομή αρχείου, δηλαδή όταν το JTextField πάνω από τα κουμπιά είναι άδειο,
 - ii. έχει εισαχθεί λάθος όνομα αρχείου,
 - iii. ολοκληρωθεί επιτυχημένα το άνοιγμα αρχείου.
 - c.

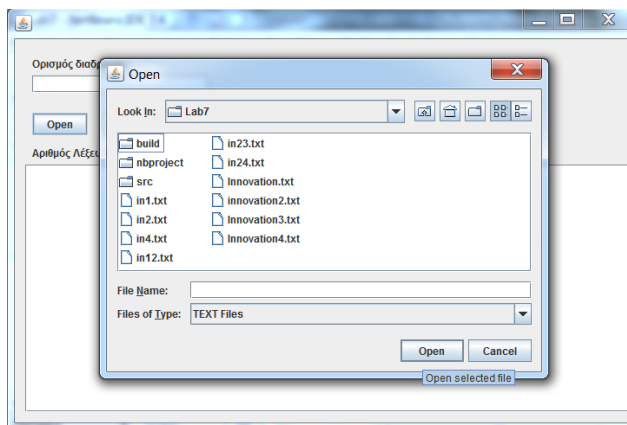
```
fw = new FileWriter(όνομα αρχείου και διαδρομή);
jTextArea1.write(fw);
fw.close();
```

Χρησιμοποιήστε τον παραπάνω κώδικα για να γράψετε σε αρχείο.
 4. Να εμφανίζετε το συνολικό αριθμό των χαρακτήρων που περιέχει το TextArea.
 5. Να εμφανίζετε το συνολικό αριθμό των λέξεων που περιέχει το TextArea.
 - a. Για να «κόψετε» τις λέξεις θα χρησιμοποιήσετε τον παρακάτω κώδικα:

```
String [] strsplit = str.split("\\W+");
```

όπου str είναι το String όπου έχετε «βάλει» το κείμενο που περιέχει το TextArea.
 6. Δημιουργήστε αντίγραφο της εφαρμογής σας επιλέγοντας Refactor – Copy από το pop-up menu που ενεργοποιείται με το δεξί πλήκτρο του ποντικιού πάνω στο όνομα του αρχείου.
 7. Τροποποιήστε την λειτουργία του open ώστε η επιλογή του αρχείου κειμένου που θα εμφανίσει ο Editor σας να γίνεται με χρήση του JFileChooser.





Χρήσιμες μέθοδοι και τάξεις για την JFileChooser

- Τάξη FileNameExtensionFilter δημιουργεί φίλτρο για τα εισαγόμενα αρχεία.
- Μέθοδος setFileFilter() ορίζει φίλτρο σε JFileChooser.
- Μέθοδος showOpenDialog() ανοίγει JFileChooser.
- Μέθοδος getSelectedFile().getName() επιστρέφει το όνομα του επιλεγμένου αρχείου.
- Μέθοδος getCurrentDirectory() επιστρέφει το όνομα του τρέχων καταλόγου.

Ανάλυση και εξήγηση λύσης

Βήμα 1: Δημιουργούμε ένα νέο project στο NetBeans με όνομα Askisi9.

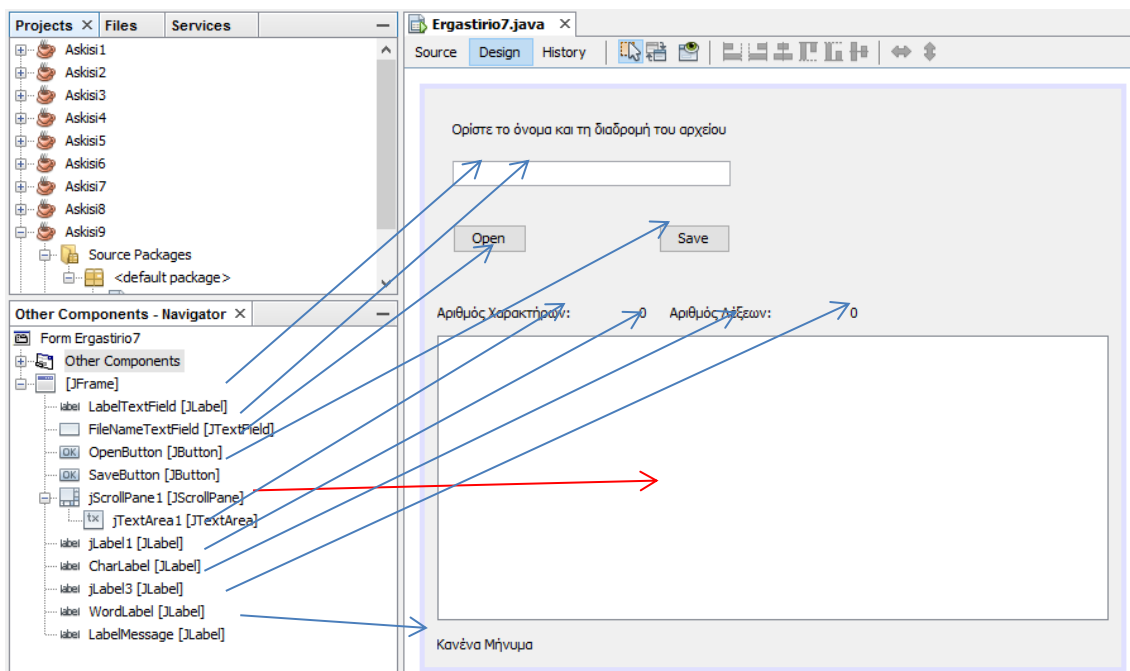
Βήμα 2: Δημιουργούμε ένα νέο Java αρχείο τύπου JFrame Form με όνομα Ergastirio7.

Βήμα 3: Ορίζουμε ως τίτλο του JFrame το «simple editor».

Βήμα 4: Πρώτα θα ασχοληθούμε με τη σχεδίαση της εφαρμογής. Έτσι, προσθέτουμε στην εφαρμογή τα παρακάτω συστατικά (Εικόνα 1):

1. **JLabel.** Όνομα Μεταβλητής: LabelTextField
Τίτλος: Ορίστε το όνομα και τη διαδρομή του αρχείου
2. **TextField.** Όνομα Μεταβλητής: FileNameTextField
Τίτλος: Κενό
Horizontal Size: 220
3. **JButton.** Όνομα Μεταβλητής: OpenButton
Τίτλος: Open
4. **JButton.** Όνομα Μεταβλητής: SaveButton
Τίτλος: Save
5. **JLabel.** Όνομα Μεταβλητής: jLabel1
Τίτλος: Αριθμός Χαρακτήρων

6. **JLabel.** Όνομα Μεταβλητής: CharLabel
Τίτλος: 0
Horizontal Alignment: Right
Horizontal Size: 50
7. **JLabel.** Όνομα Μεταβλητής: jLabel2
Τίτλος: Αριθμός Λέξεων:
8. **JLabel.** Όνομα Μεταβλητής: WordLabel
Τίτλος: 0
Horizontal Alignment: Right
Horizontal Size: 50.
9. **JTextArea.** Όνομα Μεταβλητής: jTextArea1
10. **JLabel.** Όνομα Μεταβλητής: LabelMessage
Τίτλος: Κανένα Μήνυμα



Εικόνα 1: Η σχεδίαση της εφαρμογής

Βήμα 5: Σε αυτό το βήμα θα δημιουργήσουμε τον κώδικα του γεγονότος ActionPerformed του JButton Open, όπως φαίνεται στην Εικόνα 2. Αναλυτικά:

Στη γραμμή 140: Δηλώνουμε ένα αντικείμενο fr της τάξης FileReader για να διαβάσουμε ένα αρχείο από το δίσκο. Θα χρειαστεί η βιβλιοθήκη java.io.FileReader.

Στη γραμμή 141: Αποθηκεύουμε στη μεταβλητή filename το περιεχόμενο του FileNameTextField JTextField το οποίο θα περιέχει το όνομα του αρχείου που θέλουμε να φορτώσουμε και τη διαδρομή του αρχείου.

Στη γραμμή 142: Ελέγχουμε αν δεν έχει δοθεί αρχείο και εμφανίζουμε κατάλληλο μήνυμα (γραμμή 143) στη LabelMessage JLabel.

Στις υπόλοιπες γραμμές διαβάζεται το αρχείο και εμφανίζεται στην περιοχή του JTextArea ενώ γίνεται και διαχείριση σφαλμάτων. Αναλυτικά:

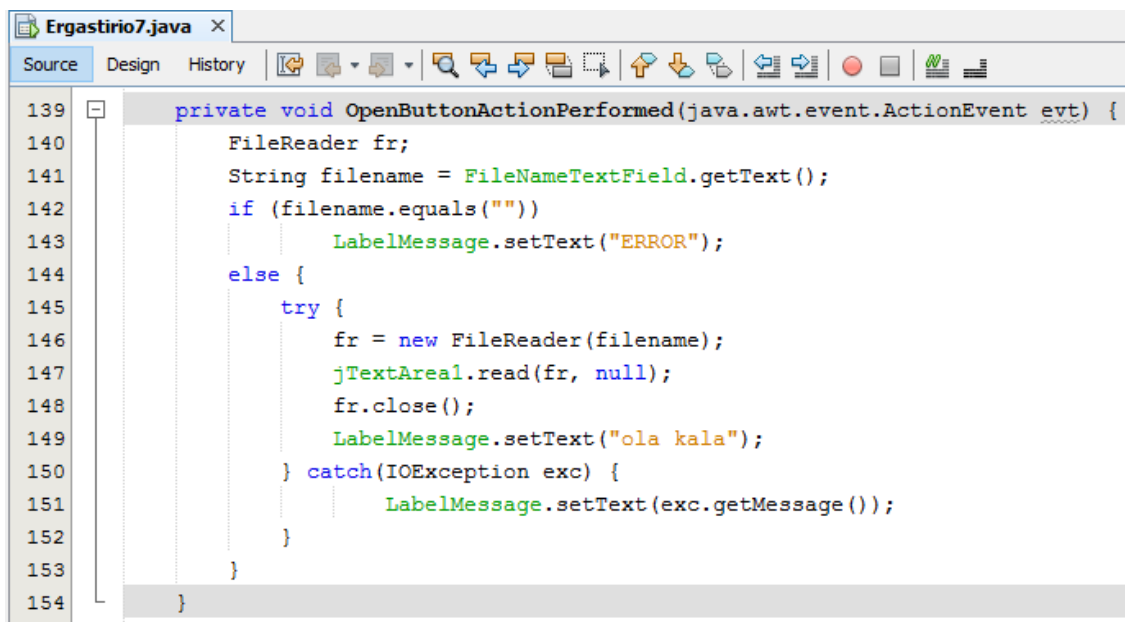
Στη γραμμή 145 και 150-152 δηλώνετε ένα try-catch το οποίο διαχειρίζεται IO Exceptions. Ότι exceptions δημιουργηθούν στην εκτέλεση της εφαρμογής θα εμφανιστούν στη *LabelMessage JLabel* με τη μέθοδο *getMessage()*. Θα χρειαστεί η βιβλιοθήκη *java.io.IOException*

Στη γραμμή 146 δημιουργείται ένα αντικείμενο για ανάγνωση αρχείου με βάση το όνομα του αρχείου που δόθηκε στο *FileNameTextField JTextField*.

Στη γραμμή 147 εμφανίζεται στην περιοχή του JTextArea το περιεχόμενο του αρχείου του αντικειμένου *fr*. Ουσιαστικά το αντικείμενο *fr* είναι μία «γέφυρα» ανάμεσα στην εφαρμογή μας και το αρχείο κειμένου που θέλουμε να ανοίξουμε.

Στη γραμμή 148 καταστρέφουμε το αντικείμενο *fr*.

Στη γραμμή 149 εμφανίζουμε κατάλληλο μήνυμα στη *LabelMessage JLabel*.



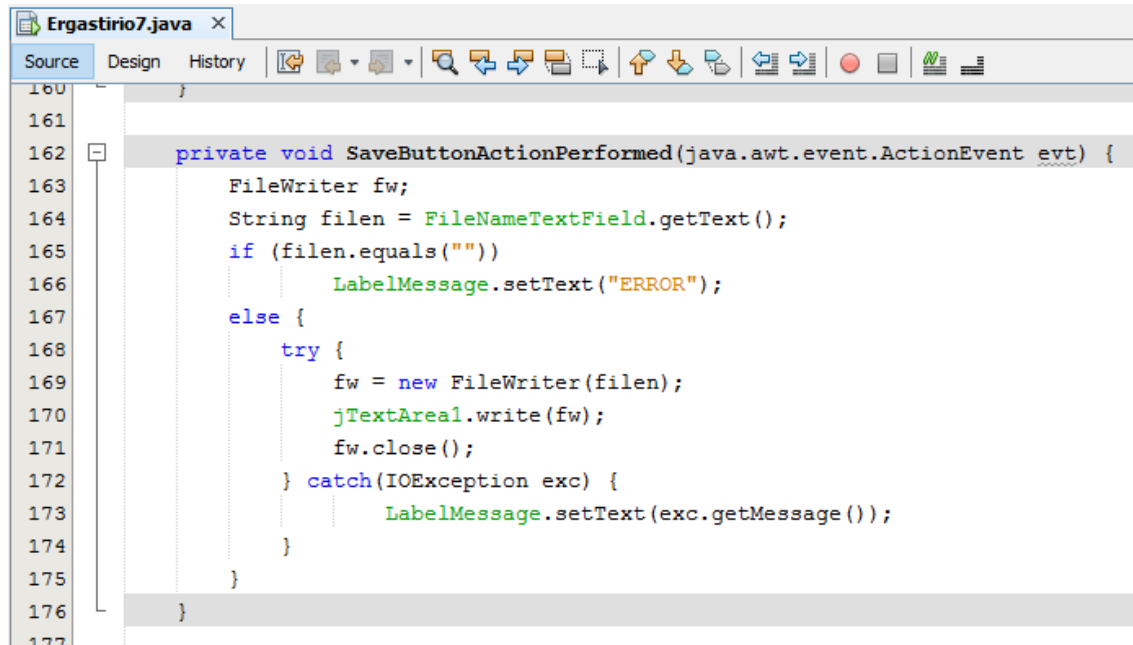
```

139 private void OpenButtonActionPerformed(java.awt.event.ActionEvent evt) {
140     FileReader fr;
141     String filename = FileNameTextField.getText();
142     if (filename.equals(""))
143         LabelMessage.setText("ERROR");
144     else {
145         try {
146             fr = new FileReader(filename);
147             JTextArea1.read(fr, null);
148             fr.close();
149             LabelMessage.setText("ola kala");
150         } catch (IOException exc) {
151             LabelMessage.setText(exc.getMessage());
152         }
153     }
154 }

```

Εικόνα 2: Ο κώδικας του Open

Βήμα 6: Σε αυτό το βήμα προγραμματίζω την αποθήκευση του περιεχομένου του JTextArea σε ένα αρχείο κειμένου, που το όνομα του αρχείου δίνεται στο *FileNameTextField JTextField*. Αν το αρχείο υπάρχει, τότε το περιεχόμενό του θα αντικατασταθεί από το περιεχόμενο του JTextArea. Αν δεν υπάρχει θα δημιουργηθεί νέο. Ο κώδικας της αποθήκευσης δίνεται στην Εικόνα 3 και είναι παρόμοιος με τον κώδικα του κουμπιού Open. Υπάρχουν μόνο δύο διαφορές. Στη γραμμή 163 και 169, δηλώνουμε και δημιουργούμε το αντικείμενο *fw* της τάξης *FileWriter*. Θα χρειαστεί η βιβλιοθήκη *java.io.FileWriter*. Στη γραμμή 170, με την μέθοδο *write(fw)* μεταφέρουμε και αποθηκεύουμε το περιεχόμενο του JTextArea στο αρχείο κειμένου που «δείχνει» το αντικείμενο *fw*.



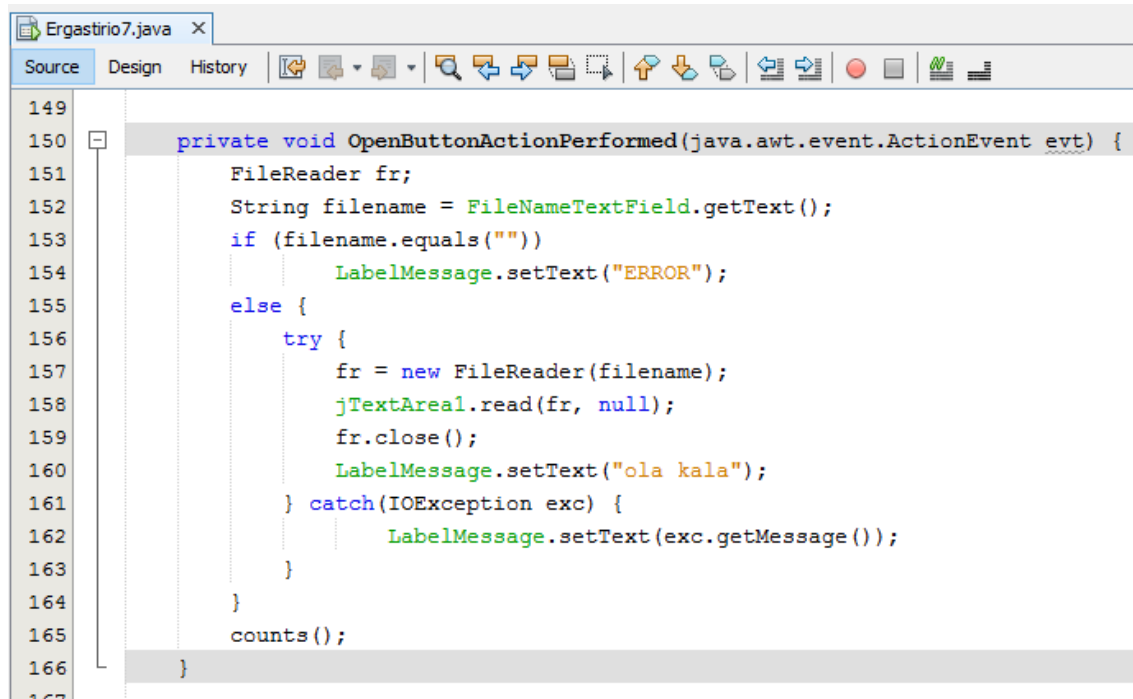
```

160 }
161
162 private void SaveButtonActionPerformed(java.awt.event.ActionEvent evt) {
163     FileWriter fw;
164     String filen = FileNameTextField.getText();
165     if (filen.equals(""))
166         LabelMessage.setText("ERROR");
167     else {
168         try {
169             fw = new FileWriter(filen);
170             JTextArea1.write(fw);
171             fw.close();
172         } catch (IOException exc) {
173             LabelMessage.setText(exc.getMessage());
174         }
175     }
176 }
177

```

Εικόνα 3: Ο κώδικας του Save

Βήμα 7: Για να εμφανίσουμε τον αριθμό των χαρακτήρων και των λέξεων που περιέχονται μέσα στο JTextArea θα πρέπει να προσέξουμε δύο σημεία της εφαρμογής μας. Πρώτον, όταν ανοίγουμε ένα αρχείο να μετράμε τους χαρακτήρες και τις λέξεις. Και δεύτερον, όταν γράφουμε ή σβήνουμε ή απλά μετακινούμε τη θέση του δείκτη του ποντικιού μέσα στο JTextArea.



```

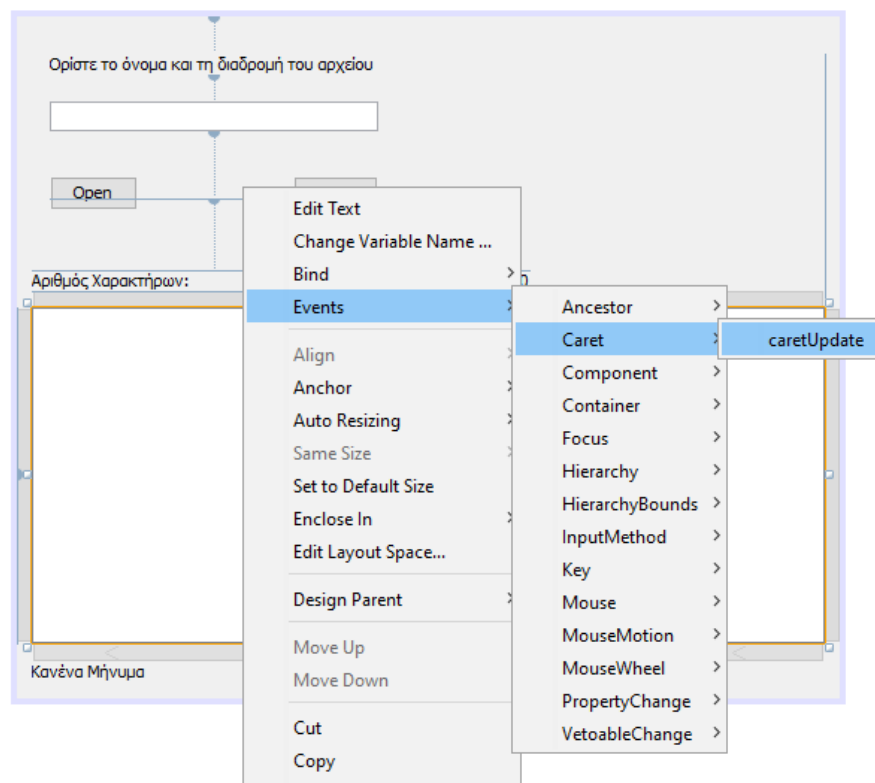
149
150 private void OpenButtonActionPerformed(java.awt.event.ActionEvent evt) {
151     FileReader fr;
152     String filename = FileNameTextField.getText();
153     if (filename.equals(""))
154         LabelMessage.setText("ERROR");
155     else {
156         try {
157             fr = new FileReader(filename);
158             JTextArea1.read(fr, null);
159             fr.close();
160             LabelMessage.setText("ola kala");
161         } catch (IOException exc) {
162             LabelMessage.setText(exc.getMessage());
163         }
164     }
165     counts();
166 }
167

```

Εικόνα 4: Στο κώδικα του Open Button προστέθηκε η κλήση της counts.

Η πρώτη περίπτωση καλύπτεται καλώντας τη μέθοδο `counts()` (μία μέθοδο που θα υλοποιήσουμε για να μετρήσουμε χαρακτήρες και λέξεις), ως τελευταία γραμμή του κώδικα που περιγράφηκε στο βήμα 5 και ο νέος κώδικας φαίνεται στην Εικόνα 4

Για τη δεύτερη περίπτωση θα πρέπει να διαχειριστούμε ένα γεγονός του `JTextArea`, το «Caret->caretUpdate» (Εικόνα 5). Εκεί θα καλέσουμε την μέθοδο `counts()`, όπως βλέπουμε στην Εικόνα 6, στις γραμμές 184-186. Στην Java Swing με τον όρο `Caret` αναφερόμαστε στον `Cursor` (Δείκτη του ποντικιού). Έτσι, το συγκεκριμένο γεγονός ενεργοποιείται όποτε μετακινείται ο δείκτης του ποντικιού στο `JTextArea` ανεξάρτητα αν το γεγονός που προκάλεσε τη μετακίνηση είναι η πληκτρολόγηση χαρακτήρων, η διαγραφή χαρακτήρων ή παραγράφων ή απλά η μετακίνηση του δείκτη.



Εικόνα 5: `JTextArea` – `Caret` event

Όπως αναφέρθηκε και παραπάνω για να εμφανίζουμε τον αριθμό των χαρακτήρων και των λέξεων, δημιουργούμε μία `void` μέθοδο με όνομα `counts()` - Εικόνα 6 (γραμμές 188 - 197). Αναλυτικά ο κώδικας:

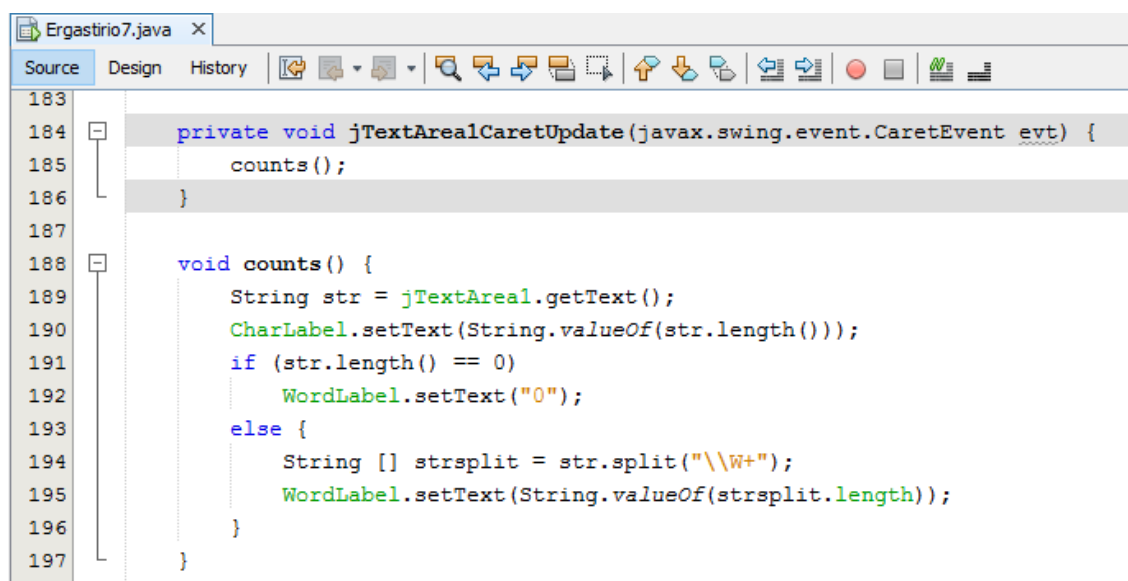
Γραμμή 189: Δηλώνουμε μία αλφαριθμητική μεταβλητή `str` στην οποία αποθηκεύουμε το περιεχόμενο του `JTextArea`.

Γραμμή 190: Εμφανίζουμε το μήκος των χαρακτήρων στην ετικέτα `CharLabel`. Το μήκος των χαρακτήρων το υπολογίζουμε με χρήση της μεθόδου `length()` στο αλφαριθμητικό `str`.

Γραμμή 191, 192: Εμφανίζουμε 0 στην ετικέτα WordLabel όταν το αλφαριθμητικό str έχει μήκος 0.

Γραμμή 193, 194: Αλλιώς, χρησιμοποιούμε τη μέθοδο split με παράμετρο «\\W+» για να αποθηκεύσουμε σε ένα πίνακα αλφαριθμητικών το περιεχόμενο της str. Ουσιαστικά, η μέθοδος split, με την συγκεκριμένη παράμετρο, όποτε συναντά στο κείμενο κενό, tab, αλλαγή γραμμής «κόβει» την λέξη που προηγείται του συμβόλου και την αποθηκεύει σε ένα κελί στον πίνακα. Έτσι, έχουμε μετατρέψει όλο το κείμενο σε ένα πίνακα αλφαριθμητικών.

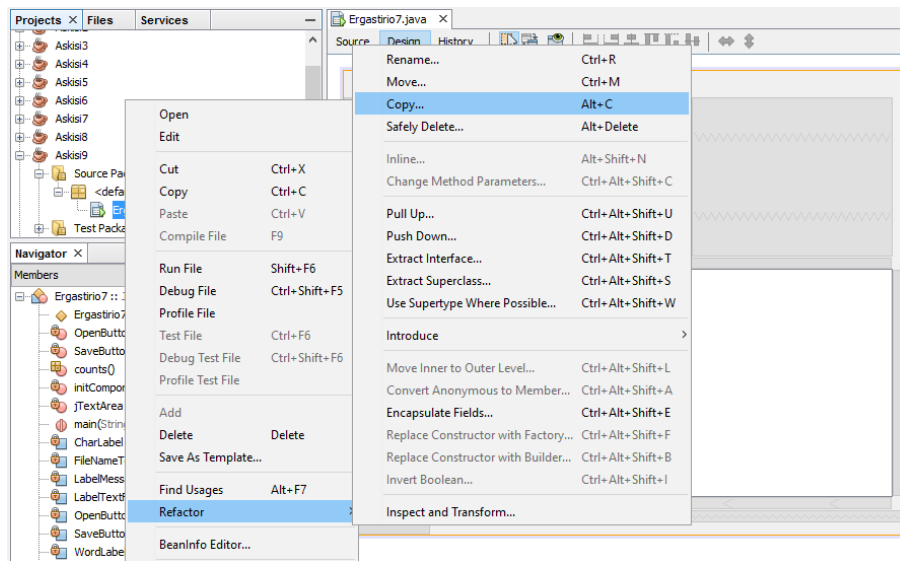
Γραμμή 195: Για να υπολογίσουμε τον αριθμό των λέξεων, αρκεί να μετρήσουμε το μήκος του πίνακα. Αυτό κάνουμε με την ιδιότητα length και εμφανίζουμε το αποτέλεσμα στην ετικέτα WordLabel.



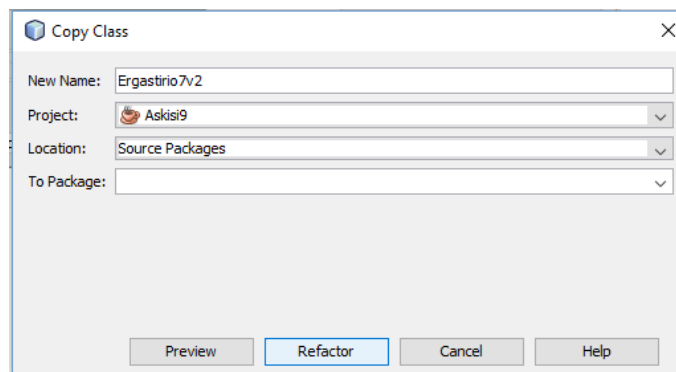
Εικόνα 6: Ο κώδικας του γεγονότος Caret και της μεθόδου counts

Βήμα 8: Στη συνέχεια θα υλοποιήσουμε την ίδια εφαρμογή με μία βελτίωση. Για το «άνοιγμα» των αρχείων θα χρησιμοποιήσουμε το συστατικό JFileChooser. Τη βελτίωση θα τη δουλέψουμε σε αντίγραφο του αρχείου που υλοποιήσαμε μέχρι τώρα.

Για να δημιουργήσουμε αντίγραφο του αρχείου κάνουμε δεξί click πάνω στο όνομα αυτού που θέλουμε να αντιγράψουμε (Ergastirio7.java) και επιλέγουμε Refactor->Copy (Εικόνα 7). Ονομάζουμε κατάλληλα το αντίγραφο (Εικόνα 8) και συνεχίζουμε την εργασία στο αντίγραφο.

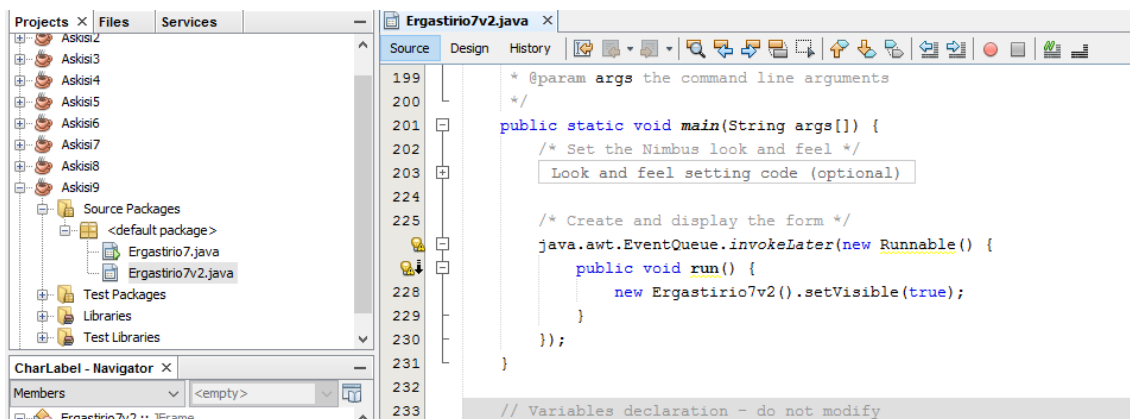


Εικόνα 7: Refactor – Copy



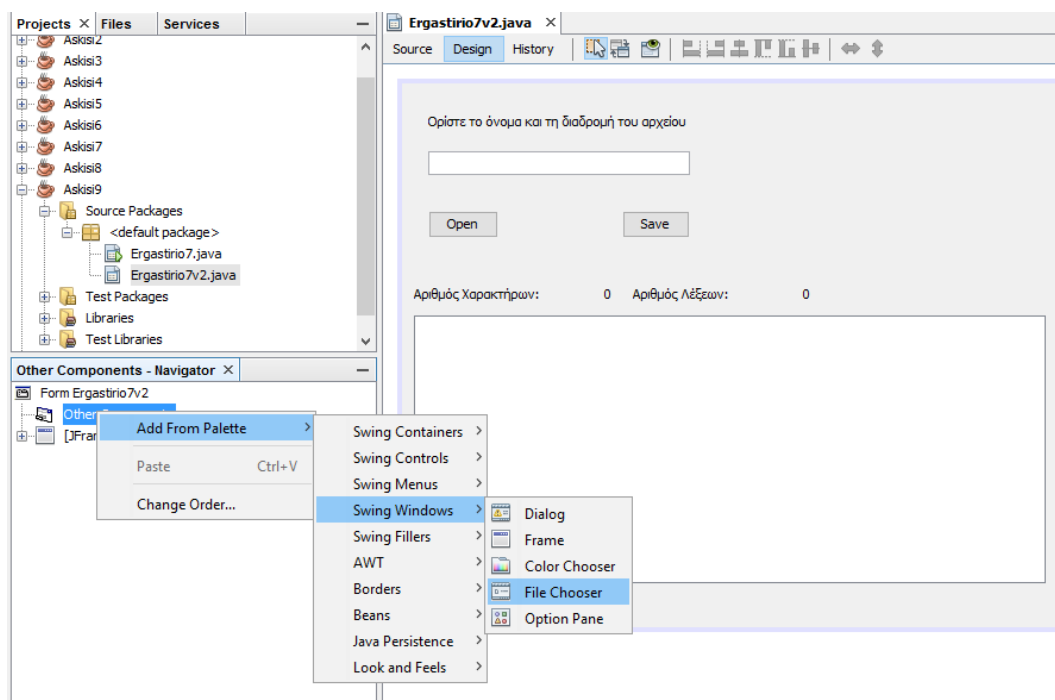
Εικόνα 8: Δίνω ένα όνομα στο νέο αντίγραφο

Προσοχή! Όταν δημιουργείται το αντίγραφο δεν αλλάζει κατάλληλα στη μέθοδο run του νήματος και η κλήση του κατάλληλου JFrame. Έτσι, θα πρέπει να τροποποιήσετε κατάλληλα τη γραμμή 228 ώστε να περιέχει το όνομα του αντίγραφου όπως φαίνεται στην Εικόνα 9.



Εικόνα 9: Αλλαγή στη main του αρχείου Ergastirio7v2

Στη συνέχεια προσθέτουμε στο design του αντίγραφου ένα JFileChooser. Θέλουμε να είναι αόρατο το JFileChooser όταν θα ξεκινήσει η εφαρμογή. Γι' αυτό το τοποθετούμε στο OtherComponents του Navigator, όπως φαίνεται στην Εικόνα 10.



Εικόνα 10: Εισαγωγή ενός File Chooser στα Other Components

Για να ανοίγει το JFileChooser όποτε πατάμε το JButton Open τροποποιούμε κατάλληλα τον κώδικα που περιγράφηκε στο βήμα 5 και στην Εικόνα 2. Ο νέος κώδικας μετά τις αλλαγές φαίνεται στην Εικόνα 11. Αναλυτικά οι αλλαγές ανά γραμμή.

Γραμμή 153. Καμία αλλαγή.

Γραμμή 154. Ανοίγουμε το jFileChooser1 με τη μέθοδο showOpenDialog() (Εικόνα 12). Η showOpenDialog ορίζει ότι το jFileChooser1 θα ανοίξει με το κείμενο (και τη λειτουργία) Open στο βασικό κουμπί του jFileChooser1. Όταν θα γίνει η τελική ενέργεια και κλείσει το jFileChooser1, θα αποθηκευτεί στην ακέραια μεταβλητή returnVal το κουμπί που έκλεισε το jFileChooser1. Δύο είναι οι τρόποι για να κλείσουμε το jFileChooser1:

1. Επιλέγοντας το κουμπί «Open» μας επιστρέφει τον ακέραιο JFileChooser.APPROVE_OPTION, όπως έχει οριστεί στην τάξη JFileChooser.
2. Επιλέγοντας το Cancel ή το κλείσιμο παραθύρου (X) μας επιστρέφει τον ακέραιο JFileChooser.CANCEL_OPTION, όπως έχει οριστεί στην τάξη JFileChooser.

Γραμμή 155. Γίνεται έλεγχος αν έχει πατηθεί το κουμπί Open.

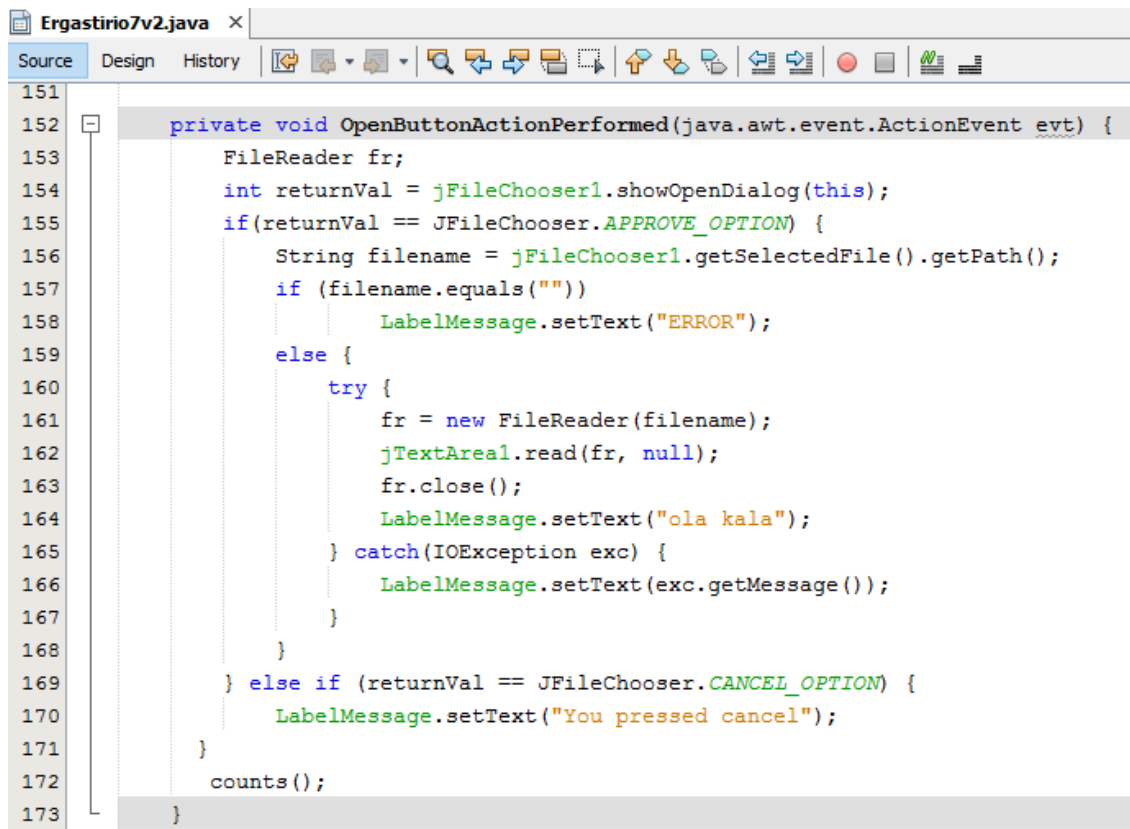
Γραμμή 156. Αποθηκεύει το όνομα του επιλεγμένου αρχείου και τη διαδρομή στην αλφαριθμητική μεταβλητή filename.

Γραμμή 157 – 168. Καμία αλλαγή.

Γραμμή 169. Γίνεται έλεγχος αν έχει πατηθεί το κουμπί Cancel ή κλείσιμο παραθύρου.

Γραμμή 170. Εμφάνιση κατάλληλου μηνύματος.

Γραμμή 172. Καμία αλλαγή.

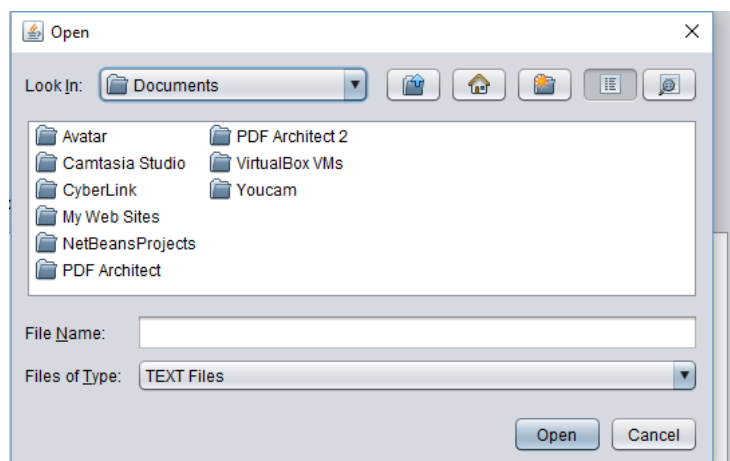


```

151
152 private void OpenButtonActionPerformed(java.awt.event.ActionEvent evt) {
153     FileReader fr;
154     int returnVal = jFileChooser1.showOpenDialog(this);
155     if(returnVal == JFileChooser.APPROVE_OPTION) {
156         String filename = jFileChooser1.getSelectedFile().getPath();
157         if (filename.equals(""))
158             LabelMessage.setText("ERROR");
159         else {
160             try {
161                 fr = new FileReader(filename);
162                 JTextArea1.read(fr, null);
163                 fr.close();
164                 LabelMessage.setText("ola kala");
165             } catch(IOException exc) {
166                 LabelMessage.setText(exc.getMessage());
167             }
168         }
169     } else if (returnVal == JFileChooser.CANCEL_OPTION) {
170         LabelMessage.setText("You pressed cancel");
171     }
172     counts();
173 }

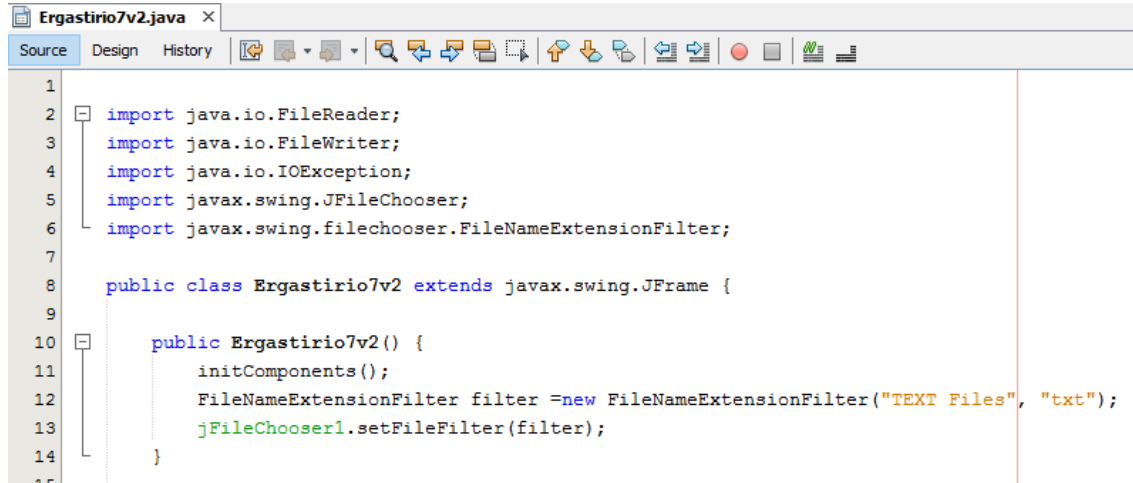
```

Εικόνα 11: Διαχείριση του JFileChooser από το κουμπί Open



Εικόνα 12: jFileChooser1

Τέλος, μπορούμε να ορίσουμε φίλτρο στο JFileChooser για να εμφανίζει μόνο συγκεκριμένα είδη αρχείων, π.χ. txt. Για να το κάνουμε αυτό ορίζουμε ένα κατάλληλο φίλτρο, όπως φαίνεται στη γραμμή 12, στην Εικόνα 13. Προσθέτουμε και την κατάλληλη βιβλιοθήκη (γραμμή 6). Στη γραμμή 13 ορίζουμε το συγκεκριμένο φίλτρο στο jFileChooser1.



Εικόνα 13: Δημιουργία φίλτρου για το JFileChooser